Assignment 3 – XD Report Template

Arushi Tyagi

CSE 13S – Winter 24

Purpose

Audience for this section: Pretend that you are working in industry, and write this paragraph for your boss. You are answering the basic question, "What does this thing do?". This section can be short. A single paragraph is okay.

Do not just copy the assignment PDF to complete this section, use your own words.

The purpose of this assignment is to create a program similar to the functions of xxd. This program is an executable called xd.c and it takes in an input that is treated as a filename. Using that input, it will read form the file and print to sdout. The contents that are printed: Index of the first byte in an 8 digit hexadecimal, hex values of the bytes that are passed, and the ASCII representation.

Questions

Please answer the following questions before you start coding. They will help guide you through the assignment. To make the grader's life easier, please do not remove the questions, and simply put your answers below the text of each question.

- What is a buffer? Why use one?
  A buffer is a storage for data and the program reads the amount of bytes designated in the buffer from a file each time read() is called. We use a buffer because if makes it easier to debug. This is because if the file is very big, we can look at it in increments and it is easier to find where the error may be when reading a certain part of the file.

- What is the return value of `read()`? What are the inputs?
  The return value of `read()` is the number of bytes that is read. The inputs are the file descriptor, buffer, and the number of bytes that is read.

- What is a file no. ? What are the file numbers of `stdin`, `stdout`, and `stderr`?
  They are integer values that are paired with each file type. The file number of stdin in 0, the file number of stdout is 1, and for stderr it's 2.

- What are the cases in which `read(0,16)` will return 16? When will it *not* return 16?
  It will return 16 when it reads 16 bytes. It will return 0 if it reads less than 16 bytes which would happen if the size of the file is less than 16 bytes.

- Give at least 2 (very differnt) cases in which a file can not be read all at once
  If the file is too large (larger than the buffer). Another case is if there is an error.

Testing

List what you will do to test your code. Make sure this is comprehensive. [1] Be sure to test inputs with delays.

[1] This question is a whole lot more vague than it has been the last few assignments. Continue to answer it with the same level of detail and thought.

Test for no arguments. The program should read from stdin.
Test for one argument. The program should take this as a file and read from that file. Then, it'll output the correct information.
Test what would happen if the user enters bytes less than 16 or over 16.

How to Use the Program

Audience: Write this section for the user of your program. You are answering the basic question, "How do I use this thing?". Don't copy the assignment exactly; explain this in your own words. This section will be longer for a more complicated program and shorter for a less complicated program. You should show how to compile and run your program. You should also describe any optional flags or inputs that your program uses, and what they do.

The user will either give an input which would be the file or the user would give no argument and enter in input using stdin. The user would keep giving input until it reaches 16 bytes and the program will print the input in its hexadecimal representation and its ASCII representation. If the user enters control d even if it is before the full 16 bytes, the program will print the bytes that have been inputted and terminate the program. If the input is not valid then the program will return a non-zero error code.

Program Design

Audience: Write this section for someone who will maintain your program. In industry you maintain your own programs, and so your audience could be future you! List the main data structures and the main algorithms. You are answering the basic question, "How is this thing organized so that I can have a chance of fixing it?". This section will be longer for a more complicated program and shorter for a less complicated program.
There will be multiple functions in the xd.c executable. The main method will check how many arguments are inputted by the user. It will track argc and argv. Another function called read_file will be in charge of reading the input/file. It will take into account the buffer size which would be 16 because it can only hold 16 bytes at a time. Depending on how large the file is, the program would call the read function enough times until the end of the file. Another file would be in charge of changing the input to hexadecimal and ascii notation. The last two functions would be called by the main function depending on how many inputs were given or if they were valid.

Pseudocode

Give the reader a top down description of your code! How will you break it down? What features will your code have? How will you implement each function. The main function would take the argc and argv arguments. Then it would check how many inputs are given based on the value of argc. If it is 1, then that means there was no input and the program would need to take the stdinput and it would call the function to calculate the ascii and hexadecimal representations. If it is 2, then it would read the file and call the readfile function which would call the ascii and hexadecimal function as well. Else, it would print an error. In the read file function, it will use the read() function and take in the buffer size. As the program is reading the file, a buffer pointer gets smaller depending on how many bytes are left in the buffer. In the hexadecimal and ascii conversion function, the program will print the input in its respective conversion using %x.

Function Descriptions

For each function in your program, you will need to explain your thought process. This means doing the following

- The inputs of every function (even if it's not a parameter)
- The outputs of every function (even if it's not the return value)

- The purpose of each function, a brief description about a sentence long.
- For more complicated functions, include pseudocode that describes how the function works
- For more complicated functions, also include a description of your decision making process; why you chose to use any data structures or control flows that you did.

Do not simply use your code to describe this. This section should be readable to a person with little to no code knowledge. **DO NOT JUST PUT THE FUNCTION SIGNATURES HERE. MORE EXPLANATION IS REQUIRED.**

- Function: Conversion. Input: buffer pointer, buffer size, index. Purpose: while the buffer pointer is not at the end of the file, the program will convert the input into ASCII and hexadecimal.
- Function: readFile. Input: the filename. Purpose: this function takes in the filename and uses the buffer to read the input 16 bytes at at time.
- Function: main. Input: arc, argv. Purpose: the function takes in the argc inputs and checks how many arguments there are. Depending on how many arguments there are, it calls the conversion and readFile functions to perform the operations.

Optimizations

This section is optional, but is required if you do the extra credit. It due `only` on your final design. You do not need it on your initial.

In what way did you make your code shorter. List everything you did!

References